COM6516: Assessed Lab 2 (20%)
Due: 11:50 on 10 December 2018

Name:	____________________
CICS User ID:	____________________

1 Rules

- **Do not turn over the page until asked to do so.**
  This sheet must not leave the lab.
- **You must work on your own.** Chat, email, message programs, and mobile phones must not be used during the assessed lab. If you talk to other students, or communicate by other means, you will receive a mark of 0.
- You can re-use *your own* code from previous lab sessions and solutions given in the module. You may also use any resources from the module web site (lecture slides; lab handouts, code, and solutions), from the official Java API and tutorials, and from textbooks and your own notes. You may use Google Translate if English is not your native language. **You may not download Java code from any other websites, or refer to any other resources.**

Module website:	`http://staffwww.dcs.shef.ac.uk/people/H.Christensen/teaching/com6516`
Java API:	`https://docs.oracle.com/javase/8/docs/api/`
Tutorials:	`https://docs.oracle.com/javase/tutorial/`

- Upload your `*.java` files to MOLE using the assignments button on the menu bar. Do not submit any other files, such as `*.class` or `*.jar` files. This assignment does not require the `sheffield` package or any libraries other than the Java standard library.)

  Your code must compile and run under Java 1.8 on the command line:

  ```
  javac *.java
  java NameOfMainClass
  ```
- When you are finished, **you must hand in this sheet** with your name and CiCS user name filled in. If you are satisfied with your work and submit before the deadline, you may leave the lab early without talking to other students. You must stop working at the end of the lab session.

  You can submit your work more than once; the last version submitted will be marked, so check it carefully. After 11:50, MOLE will record the work as late and there will be a 10% penalty. You will not be able to submit any code after 12:00.
- If you are unable to upload your code to MOLE for technical reasons, email your `*.java` files as attachments to `heidi.christensen@sheffield.ac.uk` with the subject “COM6516: Assessed Lab 2” before the deadline.
- You may ask the demonstrators questions only if you are having technical problems with the lab machines or you do not understand what is being asked for in the assignment. You may not ask programming questions.

  You can use your own laptop but the demonstrators and instructor need to be able to see your screen clearly.

2 Task

Implement a simple GUI that keeps track of the scores for participants in a competition (all scores can be assumed to be ints and < 100). The GUI allows the user to input the name of the participant and their score. It also displays who is the current top scorer. When the user quits the GUI, the full list of participants and their scores should be printed on the command line.

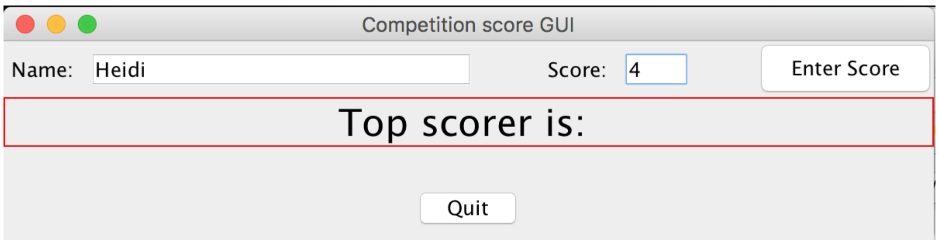


Figure 1: Example layout.

The GUI should include the following (the illustration in Figure 1 shows a possible example of the layout, but your work does not have to look exactly like it):

- two text fields for writing the participant's name and their score and a button to enter the score into the system.
- a label that displays the current top scorer's name and score. This label should be updated every time a new participant has been entered, *if* the new score is higher than the previous top score. Figure 2 shows an example of how the GUI might look as you input more and more participants - some of which become the new top scorer.
- a button with *exit* or *quit* (which should end the program and cause the full list of participants to be printed); the output printed on the command line could look something like this:

```
Name = Heidi; Score = 4
Name = Ning; Score = 10
Name = Tom; Score = 2
Name = Daisy; Score = 6
```

For the best marks, bear the following points in mind.

- There should be a top class with a main looking something like this:

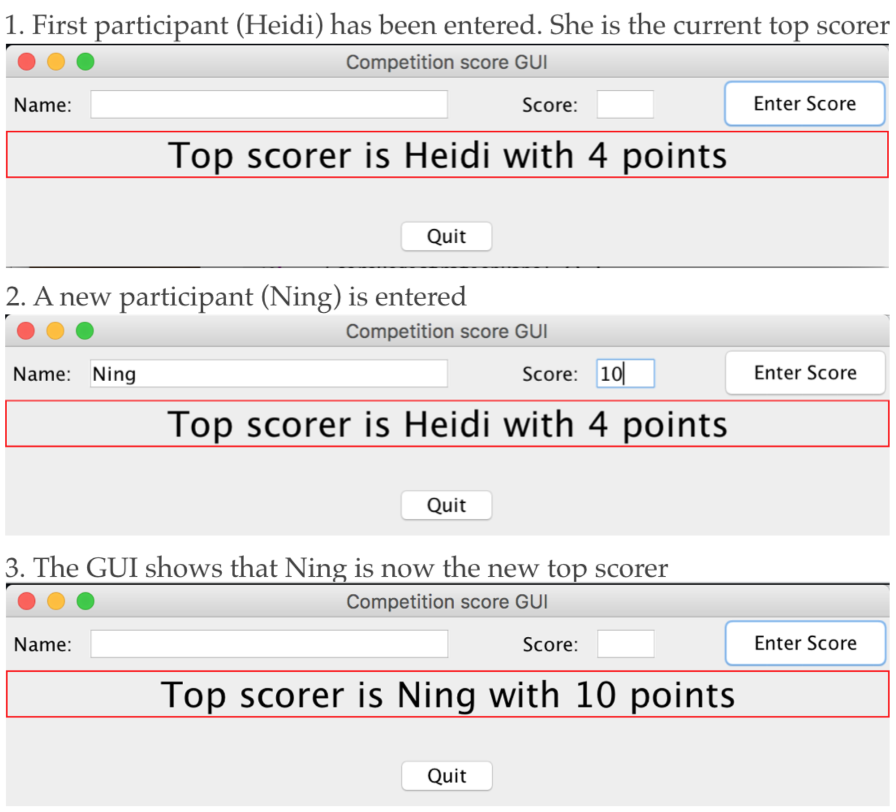
```
import javax.swing.*;


public class ScorerGUI {

    public static void main (String[] args) {
        JFrame f = new ScoringFrame ();
        f.setVisible (true);
    }

}
```

- This class should have an instance of a class extending `JFrame` which in turn should contain instances of one or more classes extending `JPanel` to get the desired layout


1. First participant (Heidi) has been entered. She is the current top scorer

2. A new participant (Ning) is entered

3. The GUI shows that Ning is now the new top scorer

Figure 2: Illustration of how the GUI might look when new participants are entered.

- You should have a separate class to be instantiated for each participant added (containing name and score), and a list of participants to which each new participant object is added.
- A listener should be used in the appropriate places.
- Each time the user creates a participant, the participant class' constructor should be called with appropriate parameters.
- Choose an appropriate class for the collection of participants, and implement the methods required for the top scorer to be maintained and for the list of participants to be printed at the end.
- The top scorer label should be distinguishable in some simple way from the other components on the GUI, for example by using a different colour or font.

3 Marking

5	compiles as specified
5	runs with the required behaviour
4	good GUI classes
4	good data representation, data structures, and supporting methods
2	style and comments
20	total

Marks and feedback will be returned through MOLE within 3 weeks.